(19)日本国特許庁（ＪＰ）　(12)公表特許公報（Ａ）　(11)特許出願公表番号
特表2001－508907
（P2001－508907A）
(43)公表日　平成13年７月３日(2001.7.3)

(51)Int.Cl.'	識別記号	ＦＩ	ﾃｰﾏｺｰﾄﾞ(参考)
Ｇ０６Ｆ　9/30	３１０	Ｇ０６Ｆ　9/30	３１０Ｅ

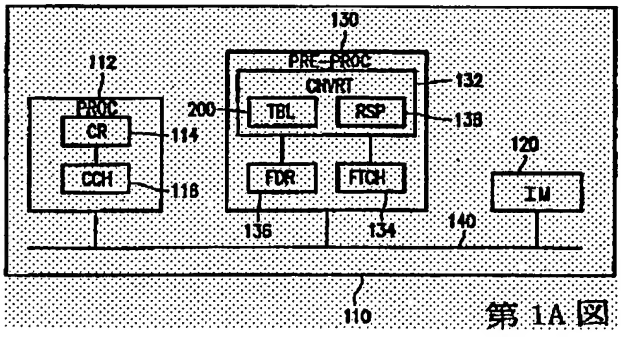
審査請求　未請求　予備審査請求　未請求(全 36 頁)

(21)出願番号　特願平11－521335
(86)(22)出願日　平成10年９月17日(1998.9.17)
(85)翻訳文提出日　平成11年６月２日(1999.6.2)
(86)国際出願番号　ＰＣＴ／ＩＢ９８／０１４３２
(87)国際公開番号　ＷＯ９９／１８４８４
(87)国際公開日　平成11年４月15日(1999.4.15)
(31)優先権主張番号　９７２０３０３３．２
(32)優先日　平成９年10月２日(1997.10.2)
(33)優先権主張国　ヨーロッパ特許庁（ＥＰ）
(81)指定国　ＥＰ(ＡＴ，ＢＥ，ＣＨ，ＣＹ，ＤＥ，ＤＫ，ＥＳ，ＦＩ，ＦＲ，ＧＢ，ＧＲ，ＩＥ，ＩＴ，ＬＵ，ＭＣ，ＮＬ，ＰＴ，ＳＥ)，ＪＰ

(71)出願人　コーニンクレッカ　フィリップス　エレクトロニクス　エヌ　ヴィ
オランダ国　5621　ベーアー　アインドーフェン　フルーネヴァウツウェッハ　1
(72)発明者　リンドワ　メンノ　メナッシェ
オランダ国　5656　アーアー　アインドーフェン　プロフ　ホルストラーン　6
(74)代理人　弁理士　沢田　雅男

(54)【発明の名称】　仮想マシン命令を実行するための処理装置

(57)【要約】
処理装置は、仮想マシン命令（例えばJavaバイトコード）を格納する命令メモリ(120)を有する。処理装置のプロセッサ(112)は、マイクロコントローラの特定命令の既定のセットから、固有命令を実行する既定のマイクロコントローラコア(14)を有する。固有命令は、仮想マシン命令とは異なる。プロセッサ(112)は、複数の固有命令の再送を要求することができるタイプである。例えば、プロセッサ(112)は、割り込みの後に再送される必要があるパイプラインおよび／または命令キャッシュを有することができる。処理装置は、命令メモリから取り出された少なくとも一つの仮想マシン命令を少なくとも一つの固有命令に変換する、コンパータ(132)を有するプレプロセッサ(130)を有する。プレプロセッサ(130)の送り手段(136)は、多くの固有命令の再送を要求するプロセッサに応答して、マイクロコントローラコア(114)に固有命令を送り出し、かつ固有命令を再送する。



第 1A 図

2004 08 16 17:01

【特許請求の範囲】

１．　仮想マシン命令を実行するための処理装置であって、前記処理装置が、

前記仮想マシン命令の少なくとも一つを含む命令を格納するための命令メモリと、

マイクロコントローラの特定の命令の既定のセットから、固有命令を実行するための既定のマイクロコントローラコアを有しているプロセッサを有していて、前記固有命令が、前記仮想マシン命令と異なっている、マイクロコントローラと、

前記命令メモリから取り込まれた少なくとも一つの仮想マシン命令を少なくとも一つの固有命令に変換するコンバータと、実行のために前記マイクロコントローラコアに固有命令を送る送り手段とを有しているプレプロセッサとを

有している処理装置において、

前記プロセッサが、割り込みのような既定の条件が発生すると、ｎ（ｎ〉１）個の固有命令の既定最大値まで再送を要求するタイプであり、そして、前記プロセッサが多くの固有命令の再送を要求すると、前記送り手段が、前記要求された固有命令を再送する手段を有する、

ことを特徴とする処理装置。

２．　前記プレプロセッサが、前記プロセッサに最後に送られた少なくともｎ個の命令を格納する送りメモリを有し、そして、

前記送り手段が、前記プロセッサが多くの命令の再送を要求すると、前記送りメモリから前記要求された命令を再送するように機能する、

ことを特徴とする請求項１に記載の処理装置。

３．　前記マイクロコントローラが、k-ステージのパイプラインのパイプラインアーキテクチャを有し、かつｎがｋ以上である、

ことを特徴とする請求項２に記載の処理装置。

４．　ｈまでの命令を格納する命令キャッシュを有しかつｎがｈ以上である、

ことを特徴とする請求項２に記載の処理装置。

5．　前記マイクロコントローラが、ｋステージパイプラインのパイプラインアーキテクチャを有し、そして、前記プロセッサが、ｈまでの命令を格納する命令キャッシュを有し、そしてｎがｋ+ｈ以上である、
ことを特徴とする請求項２に記載の処理装置。

6．　前記送りメモリが、少なくともn+m（m≧1）の固有命令を格納する位置を有し、かつ前記プレプロセッサが、前記プロセッサへの初めての送りに対しｍまでの固有命令を発生させかつそれらを前記送りメモリに格納するように機能する、
ことを特徴とする請求項２に記載の処理装置。

7．　前記プロセッサに逐次的に送られるかまたは再送される命令が、逐次的に格納され、
そして、前記プレプロセッサが、
　初めての送りに対し前記プロセッサが次に要求するであろう命令のアドレスを示すカウンタと、
　前記カウンタに格納されている前記アドレスに関して前記プロセッサにより要求された命令の実アドレスのオフセットを決定する手段とを有し、そして、前記オフセットに基づいて前記送りメモリ内の前記要求された命令をさがす、
ことを特徴とする請求項６に記載の処理装置。

8．　前記プレプロセッサが、少なくとも前記プロセッサに最後に送られた少なくともｎ個の命令を再生することが出来る前記プレプロセッサの状態を格納する格納手段を有し、
そして、前記送り手段が、前記プロセッサが、多くの命令の再送を要求すると、前記格納された状態に基づいて前記命令を再生させることにより前記要求された命令を再送するように機能する、
ことを特徴とする請求項１に記載の処理装置。

9．　前記格納手段が、前記マイクロコントローラの命令ポインタの前記状態の少なくとも一部を格納するように機能し、

そして前記送り手段が、前記命令ポインタから前記格納された部分を検索するように機能する、

ことを特徴とする請求項８に記載の処理装置。

１０．　前記プレプロセッサが、少なくとも一つの仮想マシン命令に対し、前記仮想マシン命令を固有命令のシークエンスに変換する対応する変換テーブルを有し、

前記テーブルが、固有命令および／または固有命令スケルトンを有し、

前記状態が変換テーブルオフセットインジケータを有し、

そして、前記格納手段が、前記マイクロコントローラの前記命令ポインタの既定の最下位の部分に前記変換テーブルオフセットインジケータを格納するように機能し、

そして、前記送り手段が、前記命令ポインタ内の前記変換テーブルオフセットインジケータの制御の下で、前記変換テーブル内の固有命令または固有命令スケルトンを探すように機能する、

ことを特徴とする請求項９に記載の処理装置。

１１．　前記状態が、さらに、前記命令メモリ内の仮想マシン命令を示す仮想マシン命令ポインタを有し、

そして、前記格納手段が、前記仮想マシン命令ポインタを前記マイクロコントローラの前記命令ポインタの既定の別の部分に格納するように機能し、

そして、前記プレプロセッサが、前記マイクロコントローラの前記命令ポインタ内の前記仮想マシン命令ポインタによって示される前記命令メモリ内の位置から仮想マシン命令を取り込む手段を有し、

そして、前記送り手段が、前記取り込まれた仮想マシン命令に対応する前記変換テーブルを探すように機能する、

ことを特徴とする請求項１０に記載の処理装置。

１２．　前記変換テーブルが、固有命令または固有命令スケルトンのシークエンスを有している複数のサブテーブルに、少なくともそれらの一つに、分割され、

前記変換テーブルオフセットインジケータがサブテーブルのオフセットを

示し、

前記状態が、さらに、サブテーブルインジケータを有し、

そして前記格納手段が、前記サブテーブルインジケータを前記マイクロコントローラの前記命令ポインタの既定の別の郊分に格納するように機能し、

そして、前記送り手段が、前記命令ポインタの前記サブテーブルインジケータと前記変換テーブルオフセットインジケータとによって示される前記サブテーブルの位置の固有命令または固有命令スケルトンを探すように機能する、ことを特徴とする請求項１０に記載の処理装置。

１３．　前記マイクロコントローラが、前記命令メモリおよび前記プレプロセッサを有し、

前記プロセッサ、命令メモリおよびプレプロセッサが、アトミックトランザクションマイクロコントローラバスを介して結合されていて、

前記マイクロコントローラが、前記バスを介して固有命令の（再）送を要求するタイプで、かつ前記プレプロセッサが、関連した固有命令への変換のために、前記バスを介して前記命令メモリから仮想マシン命令を取り込むように機能し、

前記プレプロセッサが、その関連する仮想マシン命令が、前記プレプロセッサに最早存在しない固有命令の（再）送を前記プロセッサが要求すると、前記プロセッサによって始められた前記バストランザクションを完了するノーオペレーション（NOP）命令を前記プロセッサに送る手段を有する、

ことを特徴とする請求項１に記載の処理装置。

１４．　マイクロコントローラの特定の命令の既定のセットから、固有命令を実行するための、既定のマイクロコントローラコアを有するプロセッサを有するマイクロコントローラと共に使用される、プレプロヤッサであって、

前記プレプロセッサが、命令メモリから取り込まれた少なくとも一つの仮想マシン命令を、少なくとも一つの固有命令に変換し、前記固有命令が前記仮想マシン命令とは異なっているコンバータと、実行のために、前記マイクロコントローラコアに固有命令を送る送り手段とを有しているプレプロセッサにおいて、

前記送り手段が、前記プロセッサが、１個以上の固有命令の再送を要求すると、前記要求された固有命令を再送する手段を有する、
ことを特徴とするプレプロセッサ。

【発明の詳細な説明】

仮想マシン命令を実行するための処理装置

技術分野

本発明は、仮想マシン命令を実行するための処理装置であって、前記処理装置が、前記仮想マシン命令の少なくとも一つを含む命令を格納するための命令メモリと、マイクロコントローラの特定の命令の既定のセットから、固有命令を実行するための既定のマイクロコントローラコアを有しているプロセッサを有していて、前記固有命令が、前記仮想マシン命令と異なっているマイクロコントローラと、前記命令メモリから取り込まれた少なくとも一つの仮想マシン命令を少なくとも一つの固有命令に変換するコンバータと、実行のために前記マイクロコントローラコアに固有命令を送る送り手段とを有しているプレプロセッサとを有している処理装置に関する。

本発明は、さらに、マイクロコントローラの特定の命令の既定のセットから、固有命令を実行するための、既定のマイクロコントローラコアを有するプロセッサを有するマイクロコントローラと共に使用される、プレプロセッサであって、

前記プレプロセッサが、命令メモリから取り込まれた少なくとも一つの仮想マシン命令を、少なくとも一つの固有命令に変換し、前記固有命令が前記仮想マシン命令とは異なっているコンバータと、実行のために、前記マイクロコントローラコアに固有命令を送る送り手段とを有しているプレプロセッサにも関する。

背景技術

ソースプログラムは、プログラムを実行するマイクロコントローラの固有命令の代わりに、ますます、仮想マシンの（コンパイルされた）命令で表されるようになって来ている。仮想マシンを使用する主な理由は、異なったマシン（プラットホーム）間のプログラムの可搬性である。仮想マシンの仮想マシン命令により表されるプログラムは、比較的簡単に、それらのマシンで実行される適切なインタープリタを使用して、いくつかの実際のマシンで実行させることができる。現

時点で、移植可能なプログラムを使用する原動力となっているものは、Javaである。Javaプログラム（アプレットと呼ばれる）は、インターネットを介して交換

され、かつ異なった命令セットを有するプロセッサを使用して、異なった固有マシンで実行させることができる。コンパイラを使用することによって、Javaアプレットは、Java仮想マシンの命令を形成するJavaバイトコード（JBCs）で表される。埋め込まれたアプリケーションに対して、仮想マシンを使用する別の原動力となっているものは、コンパクトコードのニーズである。ソフトウェアのサイズが絶えず拡大しているので、ソフトウェア圧縮技術は、以前では魅力的でなかった状況でも、ある程度の初期コストにより実用可能なものになっている。この種の手法の内の一つは、埋め込まれた特定アプリケーションに対し、固有命令よりも仮想マシン命令を用いた方がよりコンパクトにプログラムを表すことができるような適切な仮想マシンを選ぶことである。この種の仮想マシンの例は、そのコンパクトな表現で知られているスタックマシンである。埋め込まれたアプリケーションに対する特定仮想マシンは、先ず、ソースプログラムを選ばれた仮想マシン（例えばスタックマシン）の仮想マシン命令で表し、そして、さらに、コンパイル済みコードでしばしば起こる仮想マシン命令のシークエンスを、新しく定義された追加仮想マシン命令によって置換する（この場合、例えば、１つの新しい命令は、４つの既存の命令のシークエンスにより置換される）ことによって、定義することができる。

仮想マシン命令で表されるプログラムは、従来、ソフトウェア翻訳によって実行されている。プロセッサ（CPU）は、プロセッサがループ内に仮想マシン命令を取り込んで、プロセッサのマイクロコントローラコアの固有命令のシークエンスにそれをデコードし、そして各固有命令を実行する、特別なインタプリタプログラムを実行する。この手法は遅く、かつ比較的大きくなるインタプリタプログラムの追加を必要とする。実行速度を改善するためには、いわゆるJust-In-Time（JIT）コンパイル手法が、使われる。仮想マシン命令で表されるソフトウェアモジュールの始動実行の直前に、モジュールは、（固有の機械語命令で表される）固有コードにコンパイルされる。この方法の場合、モジュールは、コンパイラに対するコードに加えて二回格納されなければならない。インタプリテーション

ソフトウェアを追加して格納しなければならないことは、埋め込みシステム

2004 08 16 17:01

には望ましくない。この代わりに、ハードウェアインタプリタを使用することは望ましい。ハードウェアインタプリタ自身は、ウォーレン(Warren)の抽象命令セットに対するPrologプレプロセッサの形態で公知である。B. KnödlerおよびW. Rosenstielによる論文「ウォーレンの抽象命令セットに対するPrologプレプロセッサ」(Microprocessing and Microprogramming 18、1986年、第71-81頁）において、プレプロセッサが、Prologプログラミング言語により書かれているプログラムをモトローラ68000プロセッサ（MC68000）上でインタプリトとする点が記載されている。Prologソースプログラムを、ウォーレン氏によって定義されかつ一般にPrologプログラムを実行するために使用されている、命令に変換するためには、コンパイラが使用される。ウォーレン命令のセットは、Prologプログラムを実行するように設計されている仮想マシンを形成する。コンパイルから得られたウォーレン命令のシークエンスは、プレプロセッサを用いて、MC68000によって実行される。電源投入の後、MC68000は、固有のMC68000命令を実行することによって、先ず、ブーティング手続を実行する。ブーティング手続きが終わると、MC68000は、Prologプログラムを実行開始出来る状態になる。その実行は、既定のアドレス範囲へジャンプすることによって開始される。プレプロセッサは、この範囲にマップされているメモリマップ装置である。プレプロセッサがアドレスされると、それは、それ自身のRAMから（変換されたPrologプログラムの）ウォーレン命令を読み込み、MC68000命令および定数のシークエンスを適切に合成し、そしてこれらを実行のために直接CPUに送る。各ウォーレン命令に対するMC68000の命令は、プレプロセッサのROMに格納されている。一般に、プレプロセッサは、１つのウォーレン命令をMC68000命令のシークエンスに変換する。プレプロセッサは、プレプロセッサのRAMおよびROMの番地を生成する、それ自身のRMAコントローラおよびROMコントローラを含む。RAMコントローラは、RMA命令ポインタを管理する。連続して読み出されたMC68000の各操作により、プレプロセッサは、CPUにシークエンスの次の命令（および任意の定数）を送り出す。シークエンスが完了すると、次に、CPUに送られつつあるプログラムの次のウォーレン命令に対応するシークエンスの最初の命令が読み出され

る。割り込みが行われると、CPUの繰り返される読取り演算は、最後の命令（および任意の定数）の再送となる。

発明の開示

本発明の目的は、ある時点で複数の命令を含んでいるマイクロコントローラとの使用に適しているタイプのプロセッサ装置を提供することである。

この目的を達成するために、前記プロセッサは、割り込みのような既定の条件が発生すると、n（n > 1）個の固有命令の既定最大値まで再送を要求するタイプであり、そして、前記プロセッサが多くの固有命令の再送を要求すると、前記送り手段が、前記要求された固有命令を再送する。本発明者は、現在のプロセッサは、命令の実行がまだ開始されていないかまたはまだ完了していない状態で、一度にいくつかの固有命令を有することができることと、パフォーマンスを向上させるためには、複数の命令を含むプロセッサの能力を利用することが望ましいことを認識した。本発明者は、プロセッサが、少なくともいくつかのすでにロードされた命令を放棄し、かつ、後のステージで（例えば、割り込みのようなある条件の発生に基づいて）いくつかの命令の再送を要求する状況に対処するためには、再送メカニズムが、要求された固有命令をプロセッサに再送する必要があるとの認識を持った。

従属項２で規定される手段によると、プレプロセッサは、最後に送られた命令を格納する、FIFOのような送りメモリを有し、送りメモリから再送を行い、命令の単純で有効な再送方法を提供する。仮想マシン命令が複数の固有命令のシークエンスに変換されるという事実から、再送される第一固有命令をシークエンスの如何なる命令にもすることが出来る。さらに、固有命令は、現在のものより前の仮想マシン命令に対応させることができる。最後に、仮想マシン命令が異なっている場合には、シークエンスの長さは、異なっていてもよい。送りメモリを使用することによって、仮想マシン命令および対応する固有命令にまで戻らずに、メモリから固有命令を単純に再送することが可能になる。

従属項３において規定される手段によれば、マイクロコントローラのパイプラインで処理されつつある命令を、再送することができる。これは、プレプロセッ

サを、通常パイプラインアーキテクチャを有するRISC型プロセッサと共に使用するのに特に適している。

従属項４において規定される手段によれば、プロセッサの命令キャッシュに格納されている命令を、再送することができる。これは、プレプロセッサを、通常命令キャッシュを有するRISC型プロセッサの様な最新のプロセッサと共に使用するのに、特に適している。

従属項５において規定される手段によれば、キャッシュに格納されている命令、または、パイプラインにおいて処理されつつある命令を、再送することができる。

従属項６において規定される手段によれば、送りメモリは、再送の必要がある命令に加えて、初回に供給された命令も格納することが出来る。このような方法の場合、プレプロセッサは、固有命令が実際に要求される前に、前もって固有命令を生成させることができる。これにより、初めて要求される命令をプロセッサに供給する際に発生し得る遅延を避けることが出来る。さらに、これにより、初めて要求される命令と再送命令とを１つのみの送りメカニズムにより処理することが可能となる。

従属項７において規定される手段は、初回の送り出しと再送とを統一化する単純で有効な方法を示す。すでに送られた命令およびまだ送られていない命令は、FIFOのような送りメモリ内に逐次的に格納される。理論的には、プレプロセッサは、次に要求されるであろう命令（すなわち、まだ要求されていない最初の命令）を含む送りメモリ内のメモリーロケーションを示すポインタを維持する。実際には、このメモリーロケーションは、常に送りメモリの定位置に置くことが出来る。この結果、ポインタを離して維持する必要が無くなる。プレプロセッサは、初回の送り出しに対し次にプロセッサによって要求されるであろう固有命令のアドレスを示すカウンタを有する。通常、固有命令が初めて送られるたびに、カウンタはインクリメントされる。プロセッサは、（命令ポインタを使用して）プロセッサが次に必要とする命令のアドレスを示す。これは、命令の初回の送り出しまたはその再送の要求となる。プレプロセッサは、要求されたアドレスをカウンタに格納されているアドレスと比較する。そのオフセットに基づいて、送りメモ

リ内の命令の位置が検索される。例えば、カウンタは、最後に送られた命令

より１大きい、次に要求されるであろう命令のアドレスを格納することができる。プロセッサが、必要に応じ、この新しい命令を要求すると、オフセットはゼロとなる。論理的ポインタが新しい命令の位置を示している場合、この位置に格納されている命令がプロセッサに供給される。プロセッサが前の命令の再送を要求する場合、オフセットは、（32ビットの命令に対しバイト単位で数えて４となる一命令の単位で数えて）１または-1となる。要求された命令の位置のポインタからオフセットを減ずることによって、そのカウンタおよび論理的ポインタは、次に要求されるであろう命令の正確なアドレス／位置を格納する必要が無くなることは注目されるであろう。例えば、論理的ポインタは、最後に送られた命令も示すことができる。

従属項８において規定される手段は、送りメモリを追加する必要のない代替装置を示す。送りメモリを追加する代わりに、プレプロセッサは、プロセッサによって再送のために要求される命令を再生させることを可能にする状態情報を格納する。一例として、送られた命令を送りメモリに格納する代わりに、命令を示すポインタ、または初めに命令を生成するのに使用される情報を示すポインタをメモリに格納することもできる。

従属項９において規定される手段の場合、状態の部分はマイクロコントローラの命令ポインタに格納される。これによりプレプロセッサのコストが潜在的に減小する。

従属項10において規定される手段の場合、プレプロセッサは、少なくとも一つの仮想マシン命令に対し、仮想マシン命令を固有命令のシークエンスに変換する対応変換テーブルを有する。この変換テーブルは、例えば、ROMに格納することができる。プロセッサの命令ポインタの最下位の部分は、テーブル内のどの固有命令が必要であるかを示すために用いられる（すなわち、命令ポインタの最下位の部分は、変換テーブルオフセットインジケータとして機能する）。通常動作の間、プロセッサは命令ポインタを自動的にインクリメントさせる。プレプロセッサが、自分自身のカウンタを格納し、次に読み込まれるであろう固有命令を示す

ためにこのカウンタをインクリメントさせる必要がないことは有利である。プロセッサが再送を要求する場合、プロセッサは自動的に命令ポインタを前の値

にセットする。この方法の場合、再送も自動的に処理することができる。

従属項11において規定される手段は、変換テーブルのオフセットに加えて、要求された固有命令が位置する変換テーブルを決定するための有効な方法を提供する。命令ポインタの既定の別の部分は、命令メモリ内に仮想マシン命令を示す仮想マシン命令ポインタを有する。この仮想マシン命令ポインタは、新しい仮想マシン命令が必要となるたびにインクリメントさせる必要がある、仮想マシンプログラムカウンタと見ることができる。プレプロセッサは、命令メモリから、仮想マシンが示す仮想マシン命令を取り込み、その取り込まれた仮想マシン命令に基づいて変換テーブルを探す。次いで、プロセッサの命令ポインタの最下位の部分が、変換テーブルのオフセットとして使用される。１つの仮想マシン命令に関する固有命令のシークエンスが完了する場合はいつでも、例えば、仮想マシン命令ポインタを目標値にセットし、命令ポインタの変換オフセット部分をリセットする明示のジャンプを、シークエンス内の最後の固有命令として使用することにより、または、一つ以上のNOP（ノーオペレーション）命令を使用して、次の仮想マシン命令を示すために仮想マシン命令ポインタをセットすることができる。この結果、変換オフセット部分のオーバーフローにより、最終的に、プロセッサは、仮想マシン命令ポインタをインクリメントさせる。命令に対するプロセッサからの各要求により仮想マシン命令を命令メモリから取り込む必要が無いと言うことは理解されるであろう。プレプロセッサは、命令メモリからすでに取り込まれたまたは取り込まれるであろう仮想マシン命令を格納するキャッシュを有するのが有利である。

完全固有命令の代替として、変換テーブルが、少なくとも変数部分を有する固有命令に対して、固有命令スケルトンを含むことは有利である。スケルトンは、オペランドの様な命令の変数要素が、例えば、対応する仮想マシン命令のパラメータから最適にロードされる、演算コードの様な固有命令の既定の固定部を有する。

従属項12において規定される手段は、仮想マシン命令ポインタの使用の代替を提供する。本実施態様の場合、変換テーブルは、固有命令または固有命令スケルトンのシークエンスを有しているいくつかのサブテーブルに、（少なくともサブテーブルの一つに）、分割される。サブテーブルは、（必ずしもその必要は無いが）、１つの仮想命令に対応する固有命令の完全なシークエンスに対応させることが出来る。実際には、仮想マシン命令の固有命令の最大シークエンスより小でも良い、４つまたは８つのみの固有命令のサブテーブルのサイズを選ぶのが望ましい。この変換テーブルオフセットインジケータは、サブテーブルのうちの１つのオフセットを示す。命令ポインタの既定の別の部分は、サブテーブルインジケータを有する。プレプロセッサは、実際に関係するサブテーブルをさがすためにサブテーブルインジケータの値を使用する。好ましくは、正常動作において、固有命令が次のサブテーブルから取り込まれるたびに、（サブテーブルが、同じ仮想マシン命令に、または異なった仮想マシン命令に関しているかに関係なく）、サブテーブルインジケータは、１単位インクリメントされる。プレプロセッサは、最後に使われたサブテーブルインジケータに関連するサブテーブルへのポインタを格納することができる。特に比較的小さいサブテーブルに対しては、シークエンスがサブテーブルの境界で確実に終了するように（命令ポインタのサブテーブルインジケータ部分の値を訂正する、飛び越し命令の使用を除去するために）、１つの仮想マシン命令に関する固有命令のシークエンスについて、NOP命令によって補数をとることができる。ほとんどのプロセッサは、ジャンプ演算よりもかなり速くNOP演算を実行することができる。

従属項13において規定される手段は、プロセッサが、命令が供給されるまで命令を要求し、バスをブロック化するデッドロック状況が、発生しないことを保証する。この場合、プレプロセッサが、それが固有命令を供給することができる前に、同じバスを介して命令メモリから最初に（仮想マシン）命令を得る必要がある。

図面の簡単な説明

本発明のこれらの観点および他の観点は、図面に示されている以下の実施態様

を参照することにより、明らかになるであろう。

第１図は、処理装置内のプロセッサの可能な配置を示す。

第２図は、仮想マシン命令を固有命令のシークエンスに変換する変換テーブル

を示す。

第３図は、送りメモリが再送り命令に使用される処理装置の一実施態様のブロック図を示す。

第４図は、送り命令と再送り命令の一体化された方法を示す。

第５図は、処理装置の命令ポインタを変換テーブルへのインデックスとして使用することを示す。

第６図は、命令ポインタの別の使用を示す。

発明を実施するための最良の形態

第１図は、処理装置100内でのプレプロセッサの４つの配置の可能性を示す。処理装置100の主要コンポーネントは、マイクロコントローラ110、命令メモリ120、プレプロセッサ130である。全ての図面において、マイクロコントローラ110は、命令メモリ120とプレプロセッサ130を有している。処理装置100は、明示されていない。マイクロコントローラ110内の全ての主要コンポーネントを（望ましくは、ワンチップ素子に）結合することにより、適切な特性を得ることが出来る。必要に応じ、マイクロコントローラバス140をマイクロコントローラ110の外まで延在させ、かつ、例えば、PCIのような外部バスに結合させて、命令メモリ120および／またはプレプロセッサ130を、マイクロコントローラ110の外に置くことも出来ることは、理解されるであろう。

命令メモリ120は、スタックマシンに対する命令のような仮想マシン命令を含む。そのような仮想マシン命令の例は、Javaバイトコードである。マイクロコントローラ110は、マイクロコントローラの特定命令の既定のセットから固有命令を実行する、固有マシンと呼ばれる既定マイクロコントローラコア114を備えたプロセッサ112を有する。埋め込まれたソフトウェアを実行するのに適するマイクロコントローラの一例は、マイクロプロセッサMIPSのようなPR3001RISC型のマイクロコントローラである。マイクロコントローラコア114の固有命令は、仮想

マシンの仮想マシン命令とは異なる。このようなマイクロコントローラ110は、命令メモリ120に格納されている仮想マシン命令を直接実行することは出来ない。プロセッサ110が、命令を要求すると、プレプロセッサ130は、固有命令

を生成する。固有命令を発生させるために、プレプロセッサ130は、取り出し手段134を使用して命令メモリ120から仮想マシン命令を取り出すことが出来る。プレプロセッサ130は、命令メモリ120から取り出された仮想マシン命令を少なくとも１つの固有命令に変換するコンバータ132を有する。一般的には、固有マシン命令は、固有命令のシークエンスに変換される。プレプロセッサ130は、そのシークエンスの固有命令をその実行のためにマイクロコントローラコアに送るための送り手段136を有している。仮想マシンプログラムを実行するとき、マイクロコントローラ110は、事実、プレプロセッサ130が発生する固有プログラムを実行する。通常、マイクロコントローラ110の命令ポインタが、マイクロコントローラ110が必要とする命令メモリ120内の次の命令を次に実行することを示す場合には、その命令ポインタは、プレプロセッサに次の固有命令（または前の命令の再送）が必要であることを示す。理論的には、プレプロセッサ130が、命令メモリ120内の現在の（または次の）仮想マシン命令を示す独立の仮想マシン命令ポインタを管理する。マイクロコントローラは、仮想マシン命令または仮想マシン命令ポインタの（明示的な）知識を持たない。

プロセッサ112は、既定の条件の後、ｎ（ｎ>1）個の固有命令の既定最大値まで再送することを要求する。プロセッサに既に存在している命令を除去する割り込みを、そのような条件としても良い。ｋ個のステージでパイプライン処理できるプロセッサ112は、通常、一度にｋ個の命令を処理する。条件の発生は、全てのｋ個の命令を除去することになるかもしれない。１個以上の命令を除去する他の理由は、プロセッサが、ｈ個までの命令を格納する命令キャッシュ116を有しているいることである。条件の発生は、キャッシュ116内の全てのｈ個の命令を除去する結果になるかもしれない。後のステージで、プロセッサ112は、仮想マシンプログラムの実行に戻り１個以上の命令の再送を要求することが出来る。この目的のために、送り手段136は、プロセッサ112による多くの固有命令の再送要

求に応じて、要求された固有命令を得る手段を有する。

命令メモリ120は、固有命令も含むことが出来ることは理解されるであろう。このような固有命令は、例えば、システムを初期化する場合、またはドライバまたは埋め込まれたソフトウェアアプリケーションの特別な部分のようなある種の

ソフトウェアモジュールを、適切な性能を得るために固有命令にコンパイルする場合に使用することが出来る。固有メモリ120内の固有命令と仮想マシン命令の区別は、既定のアドレス範囲を仮想マシン命令に割り当てて、命令のアドレスにより行うことが出来る。他の方法は、その命令が固有命令であるのか仮想マシン命令であるのかを示す特別なビットを各命令毎に用いるか、または、現在の命令のタイプを示す特別なレジスタを使用することである（この場合、タイプが変更されるたびにレジスタの内容を変更する必要がある）。プレプロセッサ130は、固有命令が、命令メモリ120から変更されていないプロセッサ112に確実に与えられるようにする。

第1A図の場合、処理装置の主要コンポーネントは、PIバスのような汎用周辺相互接続バス140を介して相互接続されている。プレプロセッサ130は、アドレスの既定範囲がプレプロセッサに割り当てられている、メモリマップ周辺装置として機能することができる。プロセッサ110が、この範囲でアドレスを有する命令の要求をバス140に生成すると、プレプロセッサ130は、バス140に固有命令140を生成する。必要に応じて、プレプロセッサ130は、バス140を介して命令メモリ120から仮想マシン命令を取り出す。

第1Bおよび1C図の場合、プレプロセッサ130は、マイクロコントローラ110と命令メモリ120との間に位置する。プレプロセッサ130が固有命令と仮想マシン命令を区別する必要がある場合、これらの構成は、命令メモリ120に格納されている固有命令の実行を遅延させることが出来る。理解を容易にするために、第1B、1Cおよび1D図には、第1A図の全ての部分は、示していない。

第1D図の場合、プレプロセッサ130は、マイクロコントローラ110に埋め込まれている。プレプロセッサ130は、マイクロコントローラ110の命令キャッシュ116とコア114との間に位置するのが望ましい。この構成により、適切な性能が得ら

2004 08 16 17:01

れるが、第1B、1Cおよび1D図の構成とは異なり、マイクロコントローラ110に修正を要し、かつプレプロセッサ130は、特別仕様となるので、異なったタイプのマイクロコントローラに使用することは出来ない。

コンバータ132は、命令メモリ120から取り出された仮想マシン命令を少なくとも１つの固有メモリに変換するのに使用される。一例として、整数加算(ox60)

に対するJavaのバイトコード（仮想マシン命令）は、スタックの２つのトップ要素を加え、そのスタックからそれらのトップ要素を除き、そしてその合計をスタックにプッシュすると言う結果になる。この仮想マシン命令は、MIPSプロセッサ（32ビットマシン）に対する次に示す命令（固有命令）のシークエンスに変換させることが出来る。ここで、$tospは、（スタックのトップより上に在る）スタックの最初の空の位置を示すレジスタである。

```
lw $a0, 4 ($tosp)    /* スタックのトップ要素をレジスタ$a0にロード
lw $a1, 8 ($tosp)    /* スタックの第二要素をレジスタ$a1にロード
add $a0, $a1, $a0    /*  $a0と$a1を加え、その合計を$a0に置く
addi $tosp, $tosp,  4 /* スタックを１要素下げる
sw $a0, 4 ($tosp)    /* 合計をスタックの新しいトップに格納する
```

コンバータ132は、仮想マシン命令を固有命令のシークエンスに変換するテーブルを有するのが望ましい。テーブルの各セルが、１つの対応する仮想マシン命令に対する固有命令のシークエンスを有する、一次元のテーブルを使用することが出来る。セル番号は、対応する仮想マシン命令の値に対応させることが出来る。一例として、Javaの整数加算(0x60)の固有命令のシークエンスをセル96（16進数で0x60）に置くことが出来る。固有命令のシークエンスの長さは、様々な仮想命令に対しかなり変化させることが出来るので、シークエンスは、シークエンスが即座に互いにフォローする、何の明示セルも持たない一次元テーブルに位置させるのが望ましい。そのような変換テーブル200は、第２図に示されており、ここでは、暗示セル境界が、点線で示されている。仮想マシン命令に対するシークエンスの位置を探すことを可能とするために、各仮想マシン命令(VMI～VMI N)に対し、変換テーブル200内の対応するシークエンスの開始点を示す、コードインデ

ックステーブル210を使用することが出来る。VMI 3に対応する変換テーブル200のセルに対し、それに関連する固有命令NI 1～NI Mのシークエンス220が示されている。

変換の別の例が、Javaのバイトコードbipush n（バイトｎをextendするサインに使用されかつ結果をスタックのトップに置く）について与えられている。この仮想マシン命令は、２つのバイト(0x16とn)からなり、第一のバイトは、演算

を示し、第二のバイトは、パラメータｎを与える。命令は、固有MIPS命令の次のシークエンスに変換することが出来る。

```
ori $a0, $0, n      /*レジスタ$a0に定数ｎをロード  */
sll $a0, $a0, 24    /*24ビット左にシフト  */
sra $a0, $ao, 24    /*最も左のビットを複製する  */
                    /*ことにより右へ算術桁送りして、サイン拡張  */
sw $a0, 0 ($tosp)   /*スタックの新しいトップに結果を格納  */
addi $tosp, -4      /*スタックサイズをインクリメント  */
```

この例は、仮想マシン命令をパラメータ化することが出来ることを示している。ここで、少なくとも１つのオペランドが演算コードに従っている。コンバータ132は、変換テーブルを有している。ここで、固有命令は、フルコードまたは命令スケルトンのいずれかにより表される。一例として、命令addi $tosp, -4（前の例のシークエンスの最後の命令）は、可変部分は含まず、かつテーブル内に４バイトエントリとしてフルで置くことが出来る。命令ori $a0, $0, n（前の例のシークエンスの最初の命令）は可変部分を含み、かつ可変部分（n）を指定せずに、スケルトンとして、テーブル内に置くことが出来る。命令スケルトンに対するテーブル内のエントリは、フル命令と同じ幅（例えば、MIPSプロセッサに対し４バイト）で、一様なテーブルが得られる。固有命令スケルトンの指定されていない部分をどのように満たすかを示すためにそのテーブル内（または別のテーブル内）に別の情報を置くことも出来る。指定されていない部分を満たすのには、マイクロプログラミングを使用するのが有利である。その場合、その別の情報は、マイクロコードを有していても良いし、またそれを表示しても良い。フル固有命

令と同じ構造（幅および合成）を命令スケルトンに対し使用することが有利であることは理解されるであろう。しかしながら、他の構造を使用することも出来る。

仮想マシンがスタック指向マシンである場合、スタックまたはスタックの少なくともトップ要素は、マイクロコントローラ110のレジスタにマップされるのが望ましい。このような方法で、（仮想マシンスタックを有する）メモリスタックは、レジスタスタックにマップされる。最初、$r1が（スタックのトップより上に在る）メモリスタックの最初の空の位置に一致し、$r2がメモリスタックのトップを含み、そして$r3が、メモリスタックの第２の要素を含んで、レジスタ$r1、$r2および$r3がメモリスタックの３つの連続した要素を含むと仮定すると、Javaのバイトコードbipush nを、固有MIPS命令のシークエンス：

```
ori $r1, $0, n
sll $r1, $r1, 24
sra $r1, $r1, 24
```

に変換することができる。この演算の後、$r 1は、メモリスタックのトップを含む。同様に、最初、$r1が（スタックのトップより上に在る）メモリスタックの最初の空の位置に対応し、$r2がメモリスタックのトップを含み、かつ$r3がメモリスタックの第２要素を含む同じ位置から開始して、整数加算（0x60）のJavaバイトコード（仮想マシン命令）を、MIPS命令の次のシークエンス：

```
add $r3, $r2, $r3
```

に変換することができる。この演算の後、$r3はメモリスタックのトップを含む。

上記の実例の場合、メモリスタックのトップの位置（すなわち、メモリスタックのトップを含むレジスタ）は、コンバータ132のレジスタ138を使用して示すのが望ましい。コンバータは、適切な固有命令を発生させるために、Register Stack Pointer（RSP）と呼ばれるレジスタ138を使用する。固有命令のレジスターオペランドを指定するためには、マイクロプログラミングを用いるのが望ましい。レジスターオペランドはコンバータ132によって指定される必要があるので、こ

の方法の場合も、固定された固有命令も可変になった。この種のオペランドは、命令スケルトンを用いて、変換テーブル200にも格納されるのが望ましい。RSPが最初のフリーレジスタを示していると仮定すると、Javaのバイトコードbipush nは、対応するマイクロコードの制御の下に固有のMIPS命令の以下のシークエンスに変換することができる：

マイクロコード	命令
rsp=1;f_{lg}=rsp+1	ori $(rsp+1), $0, n
f_{tg}=rsp+1;f_{ao}=rsp+1	sll $(rsp+1), $(rsp+1), 24
f_{tg}=rsp+1;f_{ao}=rsp+1	sra $(rsp+1), $(rsp+1), 2

ここで、f_{tg}命令に対するターゲットレジスタで、そして、f_{ao}およびf_{al}は、それぞれ、第一および第二の命令に対する引数レジスタを示す。スタックの２つのトップ要素を加える次のJavaバイトコードiaddは、次のようなマイクロコードおよび命令となるであろう：

f_{tg}=rsp+2;f_{ao}=rsp+2;f_{al}=rsp+1;rsp+=1

add $(rsp+2), $(rsp+2), $(rsp+1)

第３図は、プレプロセッサ130が次に述べる送りメモリ300を有する本発明の一実施態様を示す。送りメモリ300は、最後にプロセッサに供給された少なくともｎ個（nは、プロセッサが再送を要求することが出来る固有命令の最大数）の命令を記憶するための部分310を有する。プロセッサ112が多くの命令の再送を要求すると、送り手段136は送りメモリ300から要求された命令を再送する。送りメモリ部分310は、FIFO機能136を有するのが望ましい。命令がプロセッサ112に初めて供給されるたびに、命令は部分310にロードされ、そして、部分310が満杯の場合、最も古い命令は除去される。ｋ段パイプラインを有するマイクロコントローラに対しては、ｎはｋ以上であることが望ましい。同様に、ｈ個までの命令を格納する命令キャッシュを有するプロセッサに対しては、ｎはｈ以上である。ｋ段パイプラインとｈ個までの命令を格納する命令キャッシュの両方の場合には、ｎはｋ+ｈ以上であることが望ましい。

本発明の別の実施態様の場合、送りメモリ300は、さらにｍ（$m \geqq 1$）個の固有

命令を記憶するメモリ位置320を有する。これにより、プレプロセッサ130が、前もってm個までの固有命令を初めてプロセッサに供給し、かつ、追加の命令を送りメモリ300にも格納させることが可能となる。送りメモリ300の部分320は、FIFO機能を有することが望ましい。命令がプロセッサ112に供給されるたびに、この命令は部分320から除去されて、部分310に挿入される。プレプロセッサ130は、部分320に命令を挿入する。送りメモリ300の２つのメモリ部分310および320は、結合されることが望ましい。これは、１つの一体化されたFIFOを使用することにより実行することができる。対応するポインタを有する周期的バッファのような他の変型例を、使用することもできる。

本発明の別の実施態様の場合、送りメモリ300からプロセッサ112に固有命令

を初めて送る場合と再送する場合に一体化された機構が使用される。送りメモリ300は、第４図に図示されるように、逐次的に、初回に送られた命令（部分320）または再送された命令（部分310）を格納する。プレプロセッサ130は、初回の送りに対して次回プロセッサによって要求されるであろう、固有命令のアドレスを示すカウンタ400を有する。カウンタ400は、期待命令ポインタと見ることができる。プレプロセッサ130は、プロセッサから実際に要求された固有命令を示す実命令ポインタ410を受け取る。これは、これから送られるであろう命令であってもよいが、すでに送られた命令であってもよい。プレプロセッサ130は、カウンタ400に格納されているアドレスに関してプロセッサが要求する命令（410）の実アドレスのオフセットを決定する手段を有する。受け取ろうとする命令ポインタと受け取られた命令ポインタを減ずるための減算器420が、示されている。受け取られた命令ポインタが、受け取られる命令ポインタから減算される場合、結果は、０からｎの範囲となる（ここで、０は、最初の新しい命令が要求されること、そして、他の値は、再送が要求されることを示す）。オフセットに基づいて、命令は、送りメモリ300に置かれ、かつプロセッサ112に供給される。位置探知は、オフセットを送りメモリ300へのポインタの値と組み合わせることによって実行するのが望ましい。プロセッサ112にまだ送られていない最初の固有命令（すなわち、通常の環境の下でこれから送られるであろう命令）を示すポインタ430

が、示されている。減算器440は、ポインタ430の値からオフセットを減ずる。結果として生じる値は、所望の命令を示している。プレプロセッサ130が、これかる受け取る命令ポインタと受け取られた命令ポインタとを比較したの後、これから得られる命令ポインタは、通常、受け取られた命令ポインタ＋１命令にセットされる（4バイトの命令に対しては、これは、カウンタ400を４インクリメントさせることを意味する）。送りメモリ300に対して選ばれる構成に応じて、命令が初めてプロセッサ112に送られるたびに、ポインタ430の値を１インクリメントする必要があるかもしれない。例えば、シフトレジスタを、命令を初めて送ると一つシフトする送りメモリとして使う場合には、このことは必要ではない。

送りメモリ300の使用の代替として、プレプロセッサ130は、プロセッサに最

後に送られた少なくともｎ個の命令を再生することを可能とするプレプロセッサの状態を格納する格納手段を有する。例えば、プレプロセッサ130は、最後に送られたｎ個の命令へのポインタ、またはこれらの命令の何れもが位置するまたは再生されることを許す他の状態情報へのポインタを格納する送りメモリ130と同様なメモリを有することが出来る。プロセッサが多くの命令の再送を要求すると、送り手段136は、格納されている状況に基づいて命令を発生させることにより要求された命令を再送するように機能する。

別の実施態様の場合、格納手段は、マイクロコントローラの命令ポインタの状態の少なくとも一部を格納するように機能する。これは、命令ポインタを状態の一部を反映している値にセットする、ジャンプ命令をマイクロコントローラに送出することによって簡単に実行させることができる。送り手段136は、命令ポインタから格納されている部分を読み出すように機能する。

第５図は、命令ポインタの状態の格納部分を示す。プレプロセッサ130は、少なくとも１つの仮想マシン命令に対し、仮想マシン命令を固有命令のシークエンスに変換する対応変換テーブルを有している。変換テーブルは、例えば、ROMに格納させることができる。変換テーブルは、前に記載したテーブルのセル220であってもよい（その例は、第２図に示されている）。この別の実施態様の場合、プロセッサの命令ポインタの最下位の部分は、テーブル内のどの固有命令（例え

ば、NI 1～NI n）が必要であるかを示すために使用される（すなわち、命令ポインタの最下位の部分は、変換テーブルオフセットインジケータとして機能する）。この部分は、例えば、５ビット長にすることが出来、この場合、別の手段を使用せずに仮想マシン命令に対して最大32の固有命令を可能とする。通常の運転時、プロセッサ112は命令ポインタを自動的にインクリメントさせる。プレプロセッサ130が、カウンタ400の様に自分自身のカウンタを格納する必要は無く、また次に読みとられる固有命令を示すためにこのカウンタをインクリメントさせる必要がないことは有利である。プロセッサが再送を要求する場合、プロセッサ112は、自動的に命令ポインタを前の値にセットする。この方法でも、再送を自動的に処理することができる。第５図は、プロセッサ112の命令ポインタ500の構造を示す。ここで、最下位ビット（LSB）から始まる最下位部分510は、変換テーブルオフセットインジケータとして使用される。プレプロセッサ130は、命令ポインタ500から変換テーブルオフセットインジケータ510を抽出するための手段540を有する（これは、同様に、部分510の位置で'1'ビットおよび他の部分に対して'0'ビットの既定のビットマスクを有する命令ポインタ500へのXOR演算を含むことが出来る）。抽出された部分は、変換テーブルへのポインタとしてまたはオフセットとして直接使用することができる。

別の実施態様の場合、命令ポインタ500の既定の別の部分520は、命令メモリ120の仮想マシン命令を示す仮想マシン命令ポインタを有する。この仮想マシン命令ポインタ520は、新しい仮想マシン命令が必要となる度にインクリメントさせる必要がある、仮想マシンプログラムカウンタと見ることができる。プレプロセッサ130が、メモリマップ周辺機器である場合、命令ポインタ500の１つ以上の最上位ビットは、プレプロセッサ130に対して確保されているアドレス範囲を示すために確保されている。原則として、残りのビットは、部分520に対して使うことができる。この部分は、仮想マシンプログラムの64KBブロックを可能とする、少なくとも16ビットの幅があることが望ましい。また、第1A図のレジスタスタックポインタ138は、プリプロセッサ130がこのポインタを維持する必要無しに、命令ポインタ内でエンコードさせることができる。

プレプロセッサ130は、仮想マシン命令ポインタ520を命令ポインタ500から抽出する手段530を有する。プレプロセッサ130は、命令メモリ120から抽出された仮想マシン命令ポインタによって示される仮想マシン命令を取り出す。取り出された仮想マシン命令に基づいて、プレプロセッサ130は変換テーブルをさがす。その際、プロセッサの命令ポインタ500の最下位部分510は、変換テーブルのオフセットとして使われる。手段540は、命令ポインタ500から最下位の部分510を抽出するために使われる。それぞれの仮想マシン命令のための変換テーブルは、１つのセルが、１つの仮想マシン命令に対応する固有命令のために確保されている、１つのシーケンシャルテーブルに結合されるのが望ましい。この種のテーブルは、既述されていて、その例200は、第２図に示されている。既述したように、インデックステーブル210は、各仮想マシン命令に対する変換テーブル200の関連した部分をさがすために使用することができる。加算手段560は、

変換テーブルオフセット（抽出手段540の出力）に、インデックステーブル200の出力（すなわち変換テーブル200のセルアドレス）を加算するために使用することができる。プレプロセッサ130は、命令メモリ120からすでに取り込まれたまたはこれから取り込まれる仮想マシン命令を格納するキャッシュ550を有するのが有利である。１つの仮想マシン命令に関する固有命令のシークエンスが完了した場合にはいつでも、例えば、仮想マシン命令ポインタを所望の値にセットしかつ命令ポインタの変換オフセット部分をリセットする明示のジャンプを、シークエンスの最後の固有命令として使用することにより、次の仮想マシン命令ポインタが示されるように仮想マシン命令ポインタをセットすることが出来る。命令ポインタの部分520は、部分510の次に有効性が高い部分である。この方法の場合、仮想マシン命令ポインタも、一つ以上のNOP（ノーオペレーション）命令を使用することによりインクリメントさせることができる。その結果、変換オフセット部分510のオーバーフローにより、最終的に、プロセッサ112が仮想マシン命令ポインタをインクリメントさせる。プロセッサに依っては、一つ以上のNOP命令をジャンプより速く実行することが出来る。固有命令に対するプロセッサ112からの各要求により、プレプロセッサ130が命令メモリ120から仮想マシン命令を取り込

2004 08 16 17:01

む結果になると言うことが必要ないことは注目される。

第６図に示される別の実施態様の場合、変換テーブル600は、サブテーブル602、604、606、608および609の様に、いくつかのサブテーブルに分割されている。各サブテーブルは、少なく一つの固有命令または固有命令スケルトンを有し、サブテーブルの少なくとも一つは、固有命令または固有命令スケルトンのシークエンスを有する。仮想マシン命令に対する固有命令のシークエンスは、サブテーブルの最初のエントリから始まるのが有利である。サブテーブルは、（必ずしも、この様にする必要は無いが）、１つの仮想マシン命令に対応する固有命令の完全なシークエンスに対応させることができる。サブテーブルは、シークエンスより短くてもよい。この場合、シークエンスは、逐次的に次に来るサブテーブルに続くことが望ましい。実際には、仮想マシン命令の固有命令の最大のシークエンスより小とすることが出来る４つか８つのみの固有命令のサブテーブルサイズを選ぶのが望ましい。並進テーブルオフセットインジケータ510は、サブテーブルのうちの１つのオフセットを示す。命令ポインタ500の所定の別の部分620は、サブテーブルインディケータを有する。プレプロセッサ130は、実際に関係するサブテーブルの位置を決めるために、サブテーブルインディケータ620の値を使用する。このために、プレプロセッサ130は、サブテーブルインディケータ620を命令ポインタ500から抽出する手段610を有する。通常の演算の場合において、サブテーブルインジケータは、固有命令が次のサブテーブル（そのサブテーブルが同じ仮想マシン命令に関するか異なるマシンに関するかに関係なく）から取り出されるたびに、１ユニット分インクリメントされることが望ましい。一例として、初回の仮想マシン命令のシーケンスが、110，111および112の各サブテーブル番号で逐次的にサブテーブルに格納されると仮定すると、次の仮想マシン命令は、サブテーブル85および86に格納され、そして、第三の仮想マシン命令はサブテーブル41および42に格納される。サブテーブルのサイズが、（2ビットの変換テーブルオフセットインジケータ510を使用する）４つのエントリである場合、（番号110を有するサブテーブルの）４つの固有命令の第１のシリーズは、例えば、サブテーブルインジケータ620により１の値を示すことが出来る。この場合、

変換テーブルオフセットインジケータは０から３までを示す。（番号111を有するサブテーブルの）４つの固有命令の第２のシリーズは、サブテーブルインジケータ620によって２の値と示されるであろう。第３のシリーズは、サブテーブルインジケータ620よって３の値と示されるであろう。第４のシリーズは（異なる仮想マシン命令に関しているが）、下位のテーブルインジケータ620の次の値４によって示される。プレプロセッサ130は、最後に使われたサブテーブルインジケータに対する関連したサブテーブルへのポインタを格納するためのキャッシュ630を有するのが望ましい。この例では、キャッシュは次のように格納する：

サブテーブルインジケータ	サブテーブル番号
1	110
2	111
3	112
4	85

サブテーブルは、等しいサイズを有するのが望ましい。その場合には、再送の目的に対し、サブテーブルインジケータ620に基づいて、キャッシュ630のサブテーブル番号をさがして、サブテーブル番号にサイズを乗算して、そして変換テーブルオフセット510を加算することによって、命令をさがすことができる。サブテーブルは、２のパワーである等しいサイズを有するのが有利である。対応する長さ（kビットインジケータは、2^kのサブテーブルサイズに対応する）を有する変換テーブルオフセットインジケータ510を使用することによって、サブテーブルインジケータは、サブテーブルの全ての命令を処理するたびに、（変換テーブルオフセットインジケータ510のオーバーフローの結果として）自動的にインクリメントされる。１つの仮想マシン命令に対する固有命令のシークエンスの終わりで、サブテーブルの境界で終了するようにシークエンスの補数をとるために付加NOP命令を使用することが出来る。この方法の場合、サブテーブルインジケータ620も、また、命令の連続したシークエンスに対する正しい値に、自動的にインクリメントされるであろう。これに代えて、変換テーブルオフセットインジケータ510をリセットし、サブテーブルインジケータ620をインクリメントさせるジャン

プ命令を使用することもできる。可変サイズサブテーブルに対しては、テーブル210と同様の別のインデックステーブルを、サブテーブル番号に基づいてサブテーブルの開始アドレスをさがすために使用することができる。実際には、非常に小さいサブテーブルインジケータ620を使用することが可能である。16までの命令の再送を要求し４つのエントリを有するサブテーブルを使用しているプロセッサに対しては、２ビットサブテーブルインジケータ620の使用で充分である。この例の場合、キャッシュ630は４つのエントリしか必要としない。

サブテーブルインジケータ620に対しては、実際のサブテーブル番号を使用することもできることは、理解されるであろう。しかしながら、この場合、例えば、異なったシークエンスが始まる場合、サブテーブルインジケータ620を正しい値に設定するために、ジャンプが必要となる。また、サブテーブルインジケータ620のサイズは、全てのサブテーブル番号を表示することが出来るように、より大きくなければならないであろう。このような構成を使用することによって、サブテーブルインジケータ620と変換テーブルオフセットインジケータ510との組

み合わせは、キャッシュ630を不必要にし、直接、テーブル600内のエントリを示すことができる。

本発明の別の実施態様の場合、マイクロコントローラ110は命令メモリ120とプレプロセッサ130を有する。プロセッサ112、命令メモリ120およびプレプロセッサ130は、アトミックトランザクションマイクロコントローラバス140を介して連結されている。アトミックトランザクションバスとは、命令を取り込むような読取り演算を実行するために、装置が以下のステップのシークエンスを実行するバスを意味する：

-装置が、そのバスに排他的アクセスを得る、

-装置が、バス上に要求を置くことにより（命令のような）データを要求する、

-装置が、他の装置がそのバス上にデータを置いた時にデータを得る、

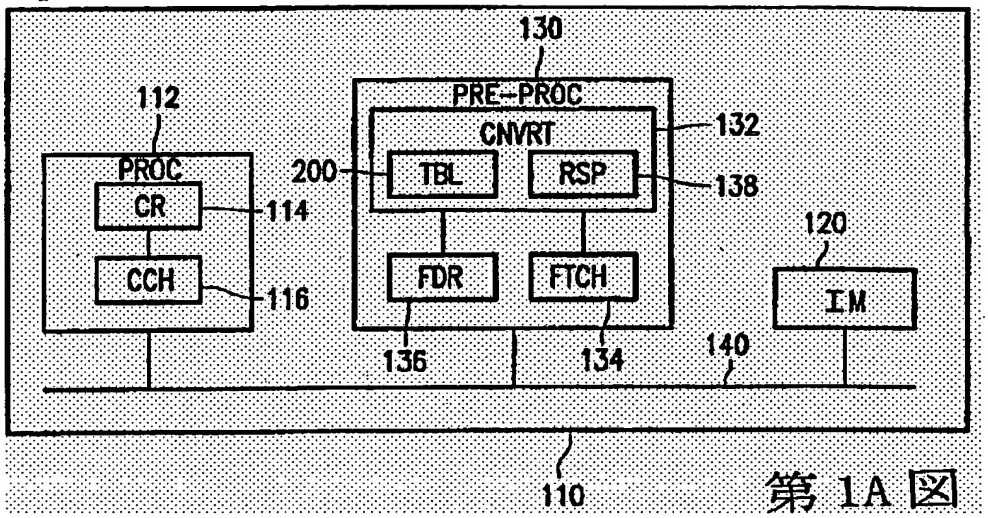
-装置が、そのバスを解放し、他の装置がそのバスにアクセスすることを可能にする。

プロセッサ112は、バス140を介して固有命令の(再)送を要求する。プレプロセッ

サ130は、関連する固有命令へ変換するために、また、バス140を介して命令メモリ120から仮想マシン命令を取り込む。特別な予防措置が無い場合、プロセッサ112が固有命令を要求し、命令が供給されるまでバス140をブロックすると、デッドロックが発生することがある。この場合、それが固有命令を供給することができる前に、プレプロセッサ130は、最初に、同じバス140を介して命令メモリ120から（仮想マシン）命令を得る必要がある。この種の状況を避けるために、プレプロセッサ130は、その関連する仮想マシン命令が、プレプロセッサ130に最早存在しない固有命令の（再）送をプロセッサ112が要求すると、プロセッサ112によって始められたバストランザクションを完了するノーオペレーション（NOP）命令をプロセッサ112に送る手段を有する。

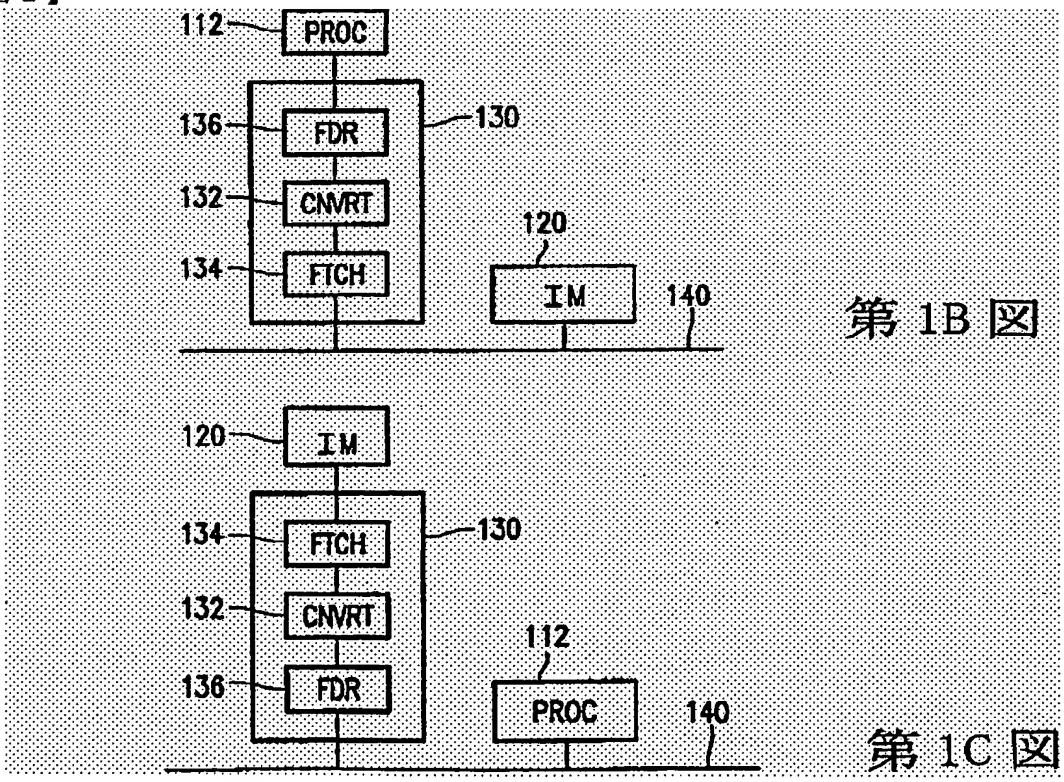
処理装置100が、複数のプログラムの仮想マシン命令を実行することが出来ることは有利である。例えば、処理装置100は、仮想マシン命令において表される複数のタスクをサポートする。オプションとして、いくつかのタスクは、固有命令で表すこともできる。多重仮想マシンタスクのサポートは、マイクロコントローラ110で実行されるオペレーティングシステムのタスク切換えルーチンによって実行されるのが望ましい。タスクスイッチが、トリガされる（例えば、タイマー割込みによって駆動されて、タスク切換え割込み処理ルーチンによって処理される）時、プロセッサ112からの情報を保存することに加えて、現在実行中のタスクに関連した情報が、プレプロセッサ130から保存される。この情報は、メモリに保存することができる。同じタイプの情報が、新しいタスクに対して再ロードされ、プレプロセッサ130およびプロセッサ112に与えられる。第３および４図に示した実施態様の場合、関連したプレプロセッサ情報は、送りメモリ300、カウンタ400およびポインタ430のコンテンツを含む。第５および６図に示した実施態様の場合、関連したプレプロセッサ情報は、キャッシュ550および630のコンテンツを、各々、含む。情報の保存と復元を可能とするために、プレプロセッサ130の関連した要素は、読取り可能でかつプロセッサ112の制御下で設定できるタイプである。代替として、キャッシュ550をプレプロセッサ130によって保存し復元することができることも理解されるであろう。

【図１Ａ】



第 1A 図

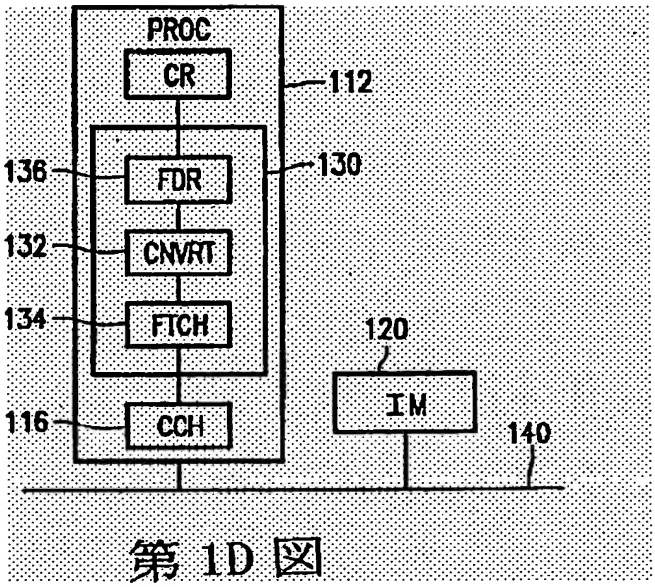
【図１】



第 1B 図

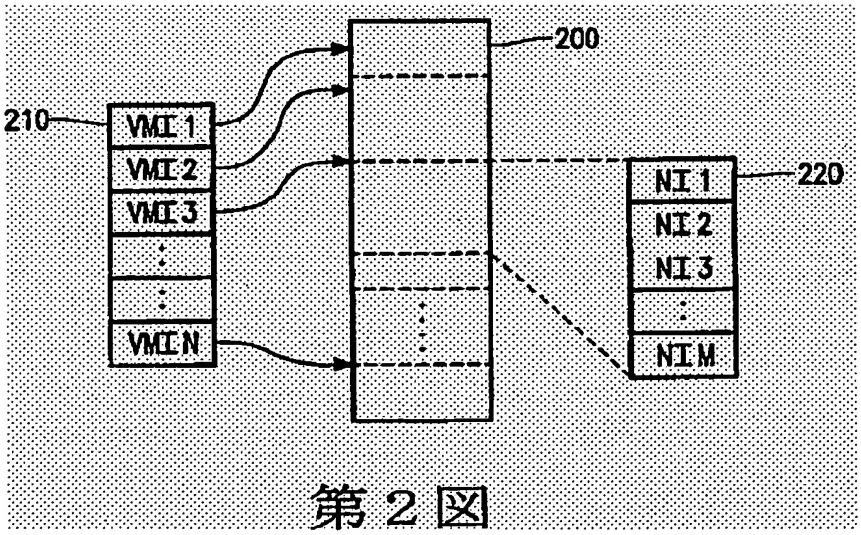
第 1C 図

【図１】



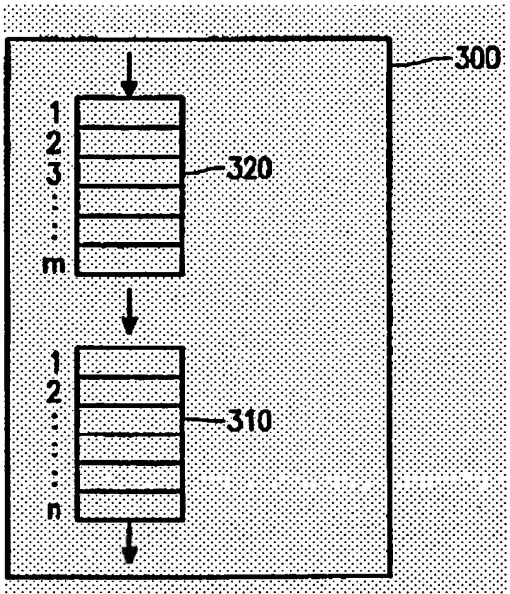
第 1D 図

【図２】



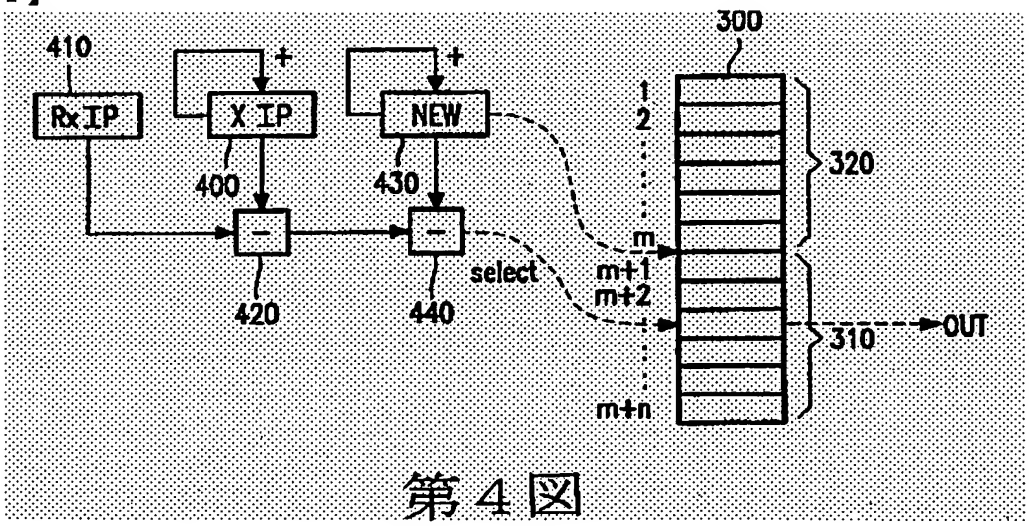
第 2 図

【図３】


300
1
2
3
m
320
1
2
n
310

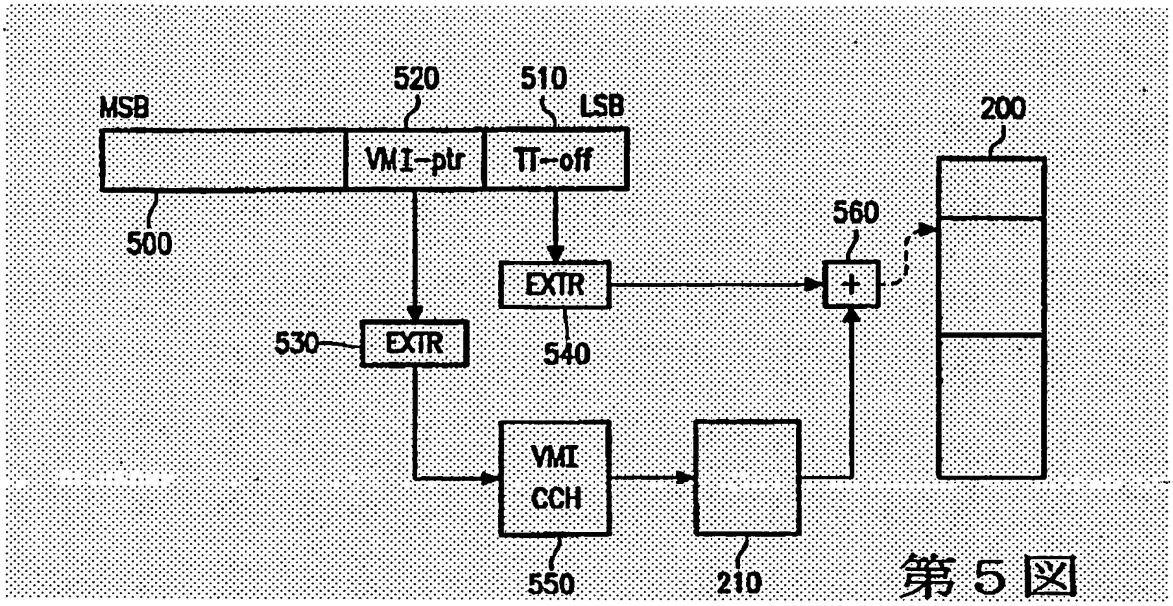
第３図

【図４】


410
RxIP
+
XIP
400
−
420
+
NEW
430
−
440
select
300
1
2
m
m+1
m+2
m+n
320
310
OUT

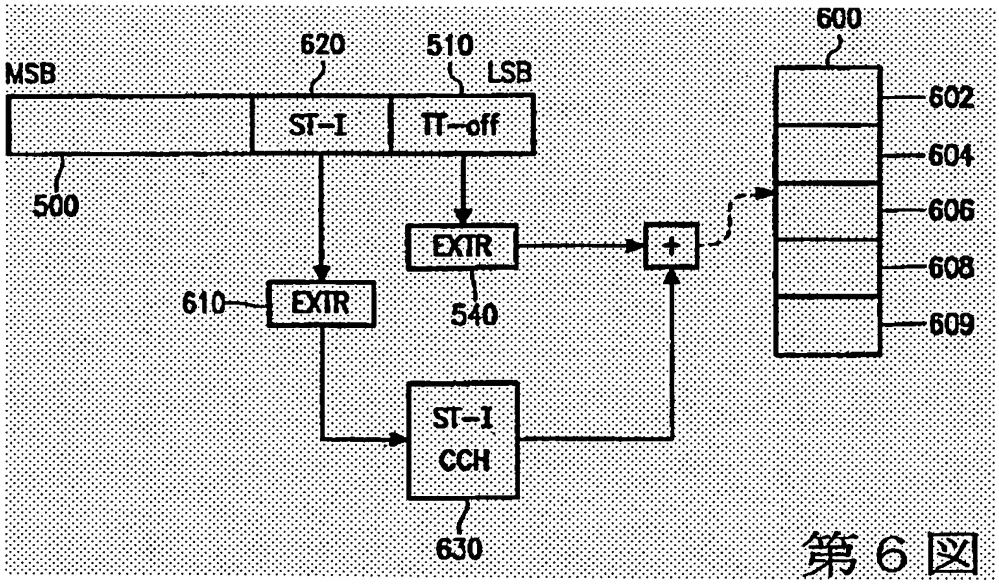
第４図

【図５】


520
510
MSB
LSB
200
VMI-ptr
TT-off
500
560
EXTR
530
EXTR
540
VMI
CCH
550
210

第５図

【図６】


600
620
510
MSB
LSB
602
ST-I
TT-off
604
500
606
EXTR
608
610
EXTR
540
609
ST-I
CCH
630

第６図

【国際調査報告】

INTERNATIONAL SEARCH REPORT

International application No.
PCT/IB 98/01432

A. CLASSIFICATION OF SUBJECT MATTER

IPC6: G06F 9/318, G06F 9/455
According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

IPC6: G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

SE,DK,FI,NO classes as above

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

WPI

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	EP 0772122 A2 (DIGITAL EQUIPMENT CORPORATION), 7 May 1997 (07.05.97), page 13, line 55 - page 14, line 24, abstract --	1-14
A	EP 0263288 A2 (INTERNATIONAL BUSINESS MACHINES CORPORATION), 13 April 1988 (13.04.88), figure 1, abstract --	1-14
A	US 4524415 A (MARVIN A. MILLS, JR. ET AL), 18 June 1985 (18.06.85), column 3, line 47 - column 4, line 10; column 5, line 36 - line 55; column 6, line 15 - line 34, abstract --	1-14

[X] Further documents are listed in the continuation of Box C. [X] See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"B" earlier document but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
12 April 1999	14-04-1999
Name and mailing address of the ISA/ Swedish Patent Office Box 5055, S-102 42 STOCKHOLM Facsimile No. +46 8 666 02 86	Authorized officer Göran Magnusson Telephone No. +46 8 782 25 00

Form PCT/ISA/210 (second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT	International application No. PCT/IB 98/01432

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	EP 0355961 A2 (DELCO ELECTRONICS CORPORATION), 28 February 1990 (28.02.90), page 5, line 34 - page 6, line 10; page 6, line 30 - line 38, figure 1, abstract ---	1-14
E,A	Patent Abstracts of Japan, abstract of JP 10-289112 A (NEC CORP), 27 October 1998 (27.10.98) ---	1-14

Form PCT/ISA/210 (continuation of second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

Information on patent family members

02/03/99

International application No.
PCT/IB 98/01432

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
EP 0772122 A2	07/05/97	AT 155905 T	15/08/97
		AU 647263 B	17/03/94
		AU 1570592 A	06/10/92
		CA 2082064 A,C	08/09/92
		DE 69221041 D,T	29/01/98
		EP 0528024 A,B	24/02/93
		FI 102219 B	00/00/00
		FI 925056 A	06/11/92
		IL 100992 A	31/12/95
		JP 7034178 B	12/04/95
		KR 9506619 B	19/06/95
		MX 9200938 A	01/03/93
		PT 100206 A,B	31/05/94
		US 5307504 A	26/04/94
		US 5432795 A	11/07/95
		WO 9215948 A	17/09/92
EP 0263288 A2	13/04/88	CA 1269756 A	29/05/90
		DE 3789345 D,T	29/09/94
		HK 79994 A	19/08/94
		JP 2022585 C	26/02/96
		JP 7058466 B	21/06/95
		JP 63098739 A	30/04/88
		SG 90794 A,G	14/10/94
		US 4841476 A	20/06/89
US 4524415 A	18/06/85	EP 0128155 A	19/12/84
		WO 8402410 A	21/06/84
EP 0355961 A2	28/02/90	CA 1316267 A	13/04/93
		DE 68925165 D,T	14/08/96
		JP 2118831 A	07/05/90
		US 5115513 A	19/05/92
		US 5117387 A	26/05/92

Form PCT/ISA/210 (patent family annex) (July 1992)